Amendments to the Claims:

This claims listing replaces all prior versions and listings of claims in the application:

Listing of Claims:

1. (Original) A method for optimizing dependencies for a set of objects comprising:

automatically detecting dependencies among a set of objects, wherein each of said objects in said set includes at least one linkable file;

adding said detected dependencies to a dependency list for said set of objects; and

removing dependencies from said dependency list for any object that does not also have at least one file dependency.

2. (Original) A method for optimizing dependencies as recited in claim 1 further comprising removing unused files from said set of objects.

3. (Currently Amended) A method for optimizing dependencies for a set of objects ~~as recited in claim 2 further~~ comprising:

automatically detecting dependencies among a set of objects, wherein each of said objects in said set includes at least one linkable file;

adding said detected dependencies to a dependency list for said set of objects;

removing dependencies from said dependency list for any object that does not also have at least one file dependency;

removing unused files from said set of objects; and

breaking a selected object in said set of objects into at least two smaller objects if said selected object is greater than a maximum object size.

4. (Original) A method for optimizing dependencies for a set of objects as recited in claim 3 wherein said threshold maximum size is a predetermined maximum object size.

5. (Original) A method for optimizing dependencies for a set of objects as recited in claim 3 further comprising making a selected file into a new object if the number of dependencies of said selected file is greater than a maximum file dependency number.

6. (Original) A method for optimizing dependencies for a set of objects as recited in claim 5 wherein said maximum file dependency number is a predetermined maximum file dependency number.

7. (Original) A method for optimizing dependencies for a set of objects as recited in claim 6 further comprising manually editing said dependency list.

8. (Original) A method for optimizing dependencies for a set of objects as recited in claim 1 further comprising manually detecting dependencies among said set of objects, and adding said manually detected dependencies to said dependency list.

9. (Original) A method for optimizing dependencies for a set of objects as recited in claim 1 wherein automatically detecting dependencies among a set of objects comprises:

recording dependencies to create a list of recorded dependencies during a traversal of said set of objects; and

analyzing said list of recorded dependencies to automatically detect dependencies.

10. (Original) A method for optimizing dependencies for a set of objects as recited in claim 1 further comprising manually editing said dependency list.

11. (Currently Amended) An apparatus for optimizing dependencies for a set of objects comprising:

means, including a processor, for automatically detecting dependencies among a set of objects, wherein each of said objects in said set includes at least one linkable file;

means for adding said detected dependencies to a dependency list for said related objects; and

means for removing dependencies from said dependency list for any object that does not also have at least one file dependency.

12. (Original) An apparatus for optimizing dependencies as recited in claim 11 further comprising means for removing unused files from said set of objects.

13. (Currently Amended) An apparatus for optimizing dependencies for a set of objects ~~as recited in claim 12 further~~ comprising:

means for automatically detecting dependencies among a set of objects, wherein each of said objects in said set includes at least one linkable file;

means for adding said detected dependencies to a dependency list for said related objects;

means for removing dependencies from said dependency list for any object that does not also have at least one file dependency;

means for removing unused files from said set of objects; and

means for breaking a selected object in said set of objects into at least two smaller objects if said selected object is greater than a maximum object size.

14. (Original) An apparatus for optimizing dependencies for a set of objects as recited in claim 13 wherein said threshold maximum size is a predetermined maximum object size.

15. (Original) An apparatus for optimizing dependencies for a set of objects as recited in claim 13 further comprising means for making a selected file into a new object if the number of dependencies of said selected file is greater than a maximum file dependency number.

16. (Original) An apparatus for optimizing dependencies for a set of objects as recited in claim 15 wherein said maximum file dependency number is a predetermined maximum file dependency number.

17. (Original) An apparatus for optimizing dependencies for a set of objects as recited in claim 16 further comprising means for manually editing said dependency list.

18. (Original) An apparatus for optimizing dependencies for a set of objects as recited in claim 11 further comprising means for manually detecting dependencies among said set of objects, and means for adding said manually detected dependencies to said dependency list.

19. (Original) An apparatus for optimizing dependencies for a set of objects as recited in claim 11 wherein said means for automatically detecting dependencies among a set of objects comprises:

means for recording dependencies to create a list of recorded dependencies during a traversal of said set of objects; and

means for analyzing said list of recorded dependencies to automatically detect dependencies.

20. (Previously Presented) An apparatus for optimizing dependencies for a set of objects as recited in claim 11 further comprising means for manually editing said dependency list.

21-33. (Canceled).